N-Terminal Switching Circuits

By E. N. GILBERT

(*Manuscript Received Feb. 14, 1951*)

The circuits considered have N accessible terminals and are operated by gangs of selector switches. Synthesis of any N-terminal switching function is accomplished. The synthesis method is proved to be economical in the sense that the switching functions which can be synthesized by any other method using much fewer contacts comprise a vanishingly small fraction of the total of all possible switching functions.

INTRODUCTION

In a recent issue of The Bell System Technical Journal[1], C. E. Shannon discussed the synthesis of two-terminal relay contact networks. Some of his results will be generalized in this paper to N-terminal networks which use selector switches with any number of positions instead of the two of a relay. The kind of circuit which will be considered may be visualized as a black box with N accessible terminals and with M shafts extending from it. Each shaft operates a selector switch (which will usually consist of several simple selector switches ganged together) inside the box. The rotors and contacts inside the box are connected electrically to one another and to the N terminals so that each way of setting the M shafts determines a pattern of interconnection of the N terminals.

We do not permit the black box to contain relay magnets or other devices which would allow the circuit to operate sequentially. Because of this restriction our results apply only to the simplest kind of switching circuit in which the state of the N terminals depends only on the present state of the M shafts, and not on the past history of the shafts. We may then use the term *N-terminal switching function* to mean a rule which assigns to each way of setting the M shafts a state of the terminals. We are concerned with the problem of synthesis: given an N-terminal switching function f, to find a switching circuit for which the states of the shafts and terminals correspond in the way indicated by f.

Let $p_1, \cdots, p_M$ be the numbers of positions which the M shafts can assume. Then there are $p_1 \cdots p_M$ different states of the shafts and the shafts have a memory[2]

[1] C. E. Shannon, *B.S.T.J.*, 28, pp. 59–98 (1949).
[2] C. E. Shannon, *B.S.T.J.*, 29, pp. 343–349 (1950).

$$H = \sum_{i=1}^{M} \log p_i$$

bits (in this paper "log" stands for "logarithm to base 2"). The results to follow include an estimate of the minimum number of contacts needed for almost all N-terminal switching functions and a network synthesis method which uses a number of contacts of the same order of magnitude as the minimum number. The number of contacts needed for almost all N-terminal switching functions is about $\frac{N \log N \, 2^H}{H + \log N}$ when H and N are large. The words "almost all" are used here in the sense that the fraction of switching functions which can be synthesized using fewer contacts than the given number tends to zero as H and/or N increases. The number of contacts used by the synthesis method is about $\frac{N^2 P 2^H}{H}$ where P is the number of positions on the largest switch. The factor P can be reduced in most cases. The analogous expressions found in Shannon's paper are $\frac{2^n}{n}$ and $\frac{2^{n+3}}{n}$ where n is the number of switching variables.

One of the most surprising facts about switching functions is that, if H is moderately large, almost none of them can be synthesized without using fantastically many contacts. This is already true of Shannon's two-terminal networks, and for N-terminal networks the situation is even worse. The reader may first turn to page 685 where a numerical example illustrating this phenomenon is given.

These paradoxical results are explained by noting that switching functions in general are much different from the usual kinds of switching functions which have practical applications. One concludes that the invention of better methods for synthesizing any imaginable function whatsoever will be of little help in practice. Almost all these functions are impossible to build (because of contact cost) and would be of no use if built. Instead one must try to isolate classes of useful switching functions which are easy to build.

Part I: Two-Terminal Networks

Selector Switches

A typical selector switch is shown in Fig. 1. It consists of a number of rotors turned by a shaft which can be set in any one of p positions. In each position of the shaft, certain of the rotors touch contacts, thereby closing those branches in the network containing the touched contacts. However, the only kinds of switches to be considered here are those with the property

that, if a contact is toucbed by a rotor wben the shaft is in position number j, then this contact remains untouched for all other positions of tbe shaft.

Networks built from two-position switches are analyzed with the aid of Boolean Algebra. It is possible to construct an algebra which is appropriate for selector switch circuits. A detailed account of this algebra bas been given by H. Piescb.[3]

The state of a switch with p positions can be associated with a switching variable x which ranges over the values 1, 2, $\cdots$, p. Then "$x = k$" means the same as "the switch is in its k^{th} position." The state of a two-terminal network, using M switches with p_1, $\cdots$, and p_M positions, is a bindrance function $f(x_1, \cdots, x_M)$ of the M switching variables $x_1, \cdots, x_M$ with x_i

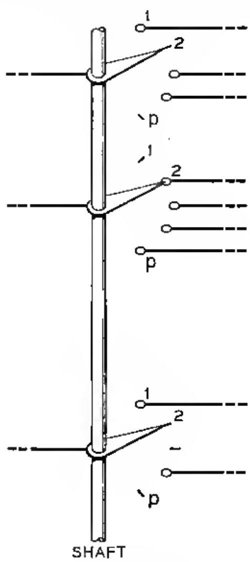

Fig. 1—Selector switch.

ranging from 1 to p_i. As usual $f = 1$ means the circuit is open and $f = 0$ means the circuit is closed. Then $f(x_1, \cdots, x_M) + g(x_1, \cdots, x_M)$ is the function representing the series connection of two networks whose functions are f and g while $f(x_1, \cdots, x_M)\, g(x_1, \cdots, x_M)$ represents the networks f and g in parallel.

The circuit which consists of just a rotor which touches a contact in its i^{th} position has hindrance function

$$e_i(x) = \begin{cases} 1 & \text{if} \quad x \neq i \\ 0 & \text{if} \quad x = i \end{cases}.$$

[3] H. Piesch, *Archiv für Electrotechnik*, 33, pp. 674, 686 and pp. 733–746 (1939).

There is no simple identity which corresponds to the Boolean Algebra expansion by sums. An identity analogous to the Boolean Algebra expansion about x_1 by products is

$$(1) \qquad f(x_1, \cdots, x_M) = \prod_{i=1}^{p} [e_i(x_1) + f(i, x_2, \cdots, x_M)]$$

where the range of x_1 is from 1 to p. To prove (1) we need only observe that in the product all the terms for which $i \neq x_1$ have the value 1. The remaining term, for which $i = x_1$, has the value $f(x_1, \cdots, x_M)$. The switching interpretation of (1) is illustrated in Fig. 2. By repeated use of (1) it follows that any function $f(x_1, \cdots, x_M)$ can be written as an expression involving

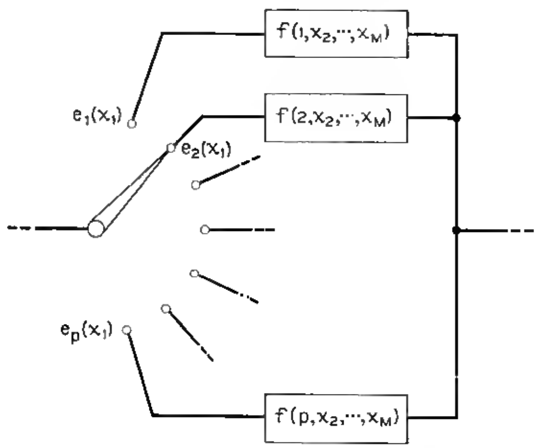

Fig. 2—Expansion of $f(x_1, \ldots, x_M)$ about x_1.

parentheses, addition signs, multiplication signs, the $e_i(x_j)$, and nothing else. Such expressions may be regarded as Boolean functions with the $e_i(x_j)$ as variables; they may be rearranged and factored according to the usual rules of Boolean Algebra. However, one should keep in mind that the $e_i(x_j)$ are subject to the constraints that a selector switch can be in only one position at any given time. The effect of these constraints is to add a cancellation law

$$e_h(x_j) + e_i(x_j) = 1 \qquad \text{if} \quad h \neq i.$$

The inverse $e_i'(x_j)$ of the Boolean variable $e_i(x_j)$ is the Boolean function

$$e_i'(x_j) = \begin{cases} 1 & \text{when} \quad e_i(x_j) = 0 \\ 0 & \text{when} \quad e_i(x_j) = 1. \end{cases}$$

Regarded as a hindrance function of the switching variable x_j,

$$e_i'(x_j) = \begin{cases} 0 & \text{when} \quad x_j \neq i \\ 1 & \text{when} \quad x_j = i \end{cases}.$$

Then by (1),

$$e_i'(x_j) = \prod_{h \neq i} e_h(x_j).$$

If the switch x_j has p_j positions, it takes $p_j - 1$ contacts to build a circuit with hindrance function $e_i'(x_j)$.

Synthesis

Suppose that M shafts, governed by switching variables $x_1, \cdots, x_M$, are given, together with a two-terminal hindrance function $f(x_1, \cdots, x_M)$. The synthesis problem is to design a network with hindrance function $f(x_1, \cdots, x_M)$, adding suitable rotors and contacts to form selector switches from the given shafts.

One solution can be found immediately:

(i) As described above, express the hindrance function $f(x_1, \cdots, x_M)$ as a Boolean function $B(\xi_1, \cdots, \xi_R)$ of Boolean variables $\xi_1, \cdots, \xi_R$ (which are the $e_i(x_j)$ with new labels). Here $R = p_1 + p_2 + \cdots + p_M$.

(ii) Any of the well known methods of synthesizing relay networks can be used to design a network operated by the Boolean variables $\xi_1, \cdots, \xi_R$ and with hindrance function $B(\xi_1, \cdots, \xi_R)$.

(iii) In the network found in (ii) replace each contact ξ_a by the appropriate $e_i(x_j)$ and each back contact ξ_a' by the appropriate circuit $e_i'(x_j)$.

The solution found in (iii) will ordinarily use up a number of contacts which is unnecessarily large by many orders of magnitude. From Shannon's theorems on relay networks we know that the probability is high, that as many as

$$\frac{2^R}{R}$$

contacts will be needed in step (ii). The final circuit (iii) will have even more contacts if some circuits $e_i'(x_j)$ are used.

The synthesis process which follows replaces the exponent $R = \sum_{i=1}^M p_i$ in the estimate of the number of contacts by the smaller number $\sum_{i=1}^M \log p_i$. The reader may recognize the process as essentially the same as the one given by Shannon for two-terminal relay networks. The network will again take the form of a tree connected to a circuit which produces all functions of the switching variables which govern it.

If one expands $f(x_1, \cdots, x_M)$ about $x_1, \cdots, x_k$ one finds

$$f = \prod_{r,s,\cdots,z} [e_r(x_1) + e_s(x_2) + \cdots + e_z(x_k) + f(r, s, \cdots, z, x_{k+1}, \cdots, x_M)] \tag{2}$$

where the multiple product is taken over the entire range of the variables $x_1, \cdots, x_k$. The different functions $e_r(x_1) + e_s(x_2) + \cdots + e_z(x_k)$ can all be realized with one tree circuit as shown in Fig. 3. If the expansion had been performed about all the variables $x_1, \cdots, x_M$, identity (2) would

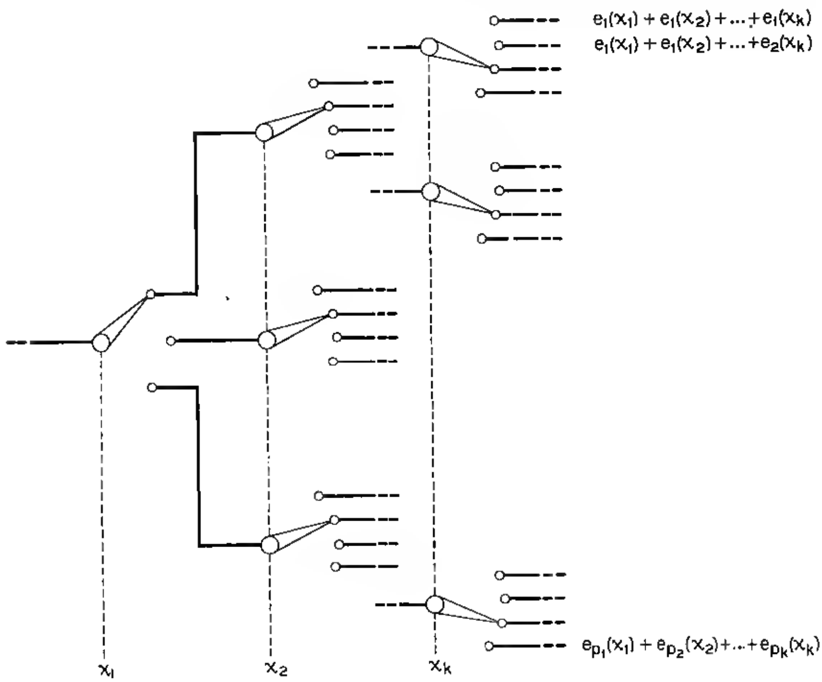

Fig. 3—Tree.

show that the desired function f could be synthesized by connecting one terminal to the input lead of a tree and the other terminal to certain of the tree's output leads. The number of contacts used would have been about 2^{H+1}. The method which follows uses still fewer contacts.

The network which we use to synthesize the function f is shown in Fig. 4. It consists of the tree of Fig. 3 with its output leads connected to the input leads of a network on the right which is designed so that the hindrances from its input leads to its output lead are the functions $f(r, s, \cdots, z, x_{k+1},$

$\cdots, x_M)$. For given values $r_0, s_0, \cdots, z_0$ of the switching variahles $x_1, \cdots, x_k$ of the tree, there is only one closed path through the tree; this path ends at the output lead lahelled $e_{r_0}(x_1) + \cdots + e_{s_0}(x_k)$.

The hindrance from this point to the output lead of the right-hand network is the hindrance of the network of Fig. 4. This hindrance is just $f(r_0, s_0, \cdots, z_0, x_{k+1}, \cdots, x_M)$ (note that the connections to the disjunctive tree do not cause any interconnections among the other leads of the right hand network), which proves that the network has the required hindrance. By proper choice of the numher k of switches in the tree we will ohtain an economical design.

Network to Produce All Functions

To produce all of $f(r, s, \cdots, z, x_{k+1}, \cdots, x_M)$ it suffices to build a circuit which produces every function of $(x_{k+1}, \cdots, x_M)$. Let these variahles he

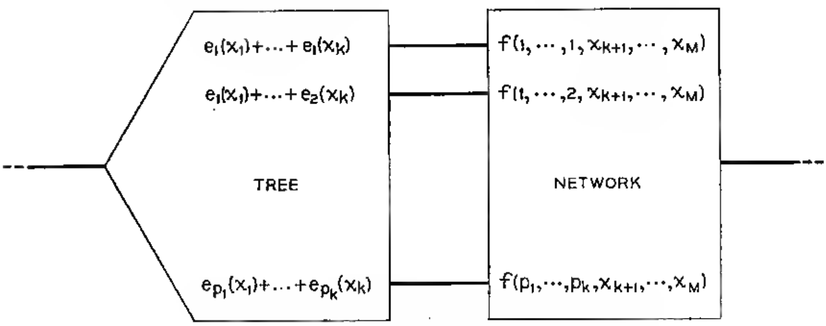

Fig. 4—Network for $f(x_1, \ldots, x_M)$.

relahelled $y_1, \cdots, y_L$ and have ranges $p_1, \cdots, p_L$. Let the largest of $p_1, \cdots, p_L$ he called P.

Theorem I. A network which produces every function of $(y_1, \cdots, y_L)$ *can be built with a number* ψ_L *of contacts satisfying*

$$\psi_L \leq P 2^{p_1 \cdots p_L}. \tag{3}$$

The proof is hy induction on L. Suppose that a network to produce every function of $(y_1, \cdots, y_{j-1})$ has heen huilt with ψ_{j-1} contacts and try to huild one for every function of $(y_1, \cdots, y_j)$. The numher of functions which the network must produce is $2^{p_1 \cdots p_j}$, for there are $p_1 \cdots p_j$ different ways of setting the switches and two choices (0 or 1) for the value of the function for each state of the switches. Of these functions, the ψ_{j-1} network itself provides $2^{p_1 \cdots p_{j-1}}$ functions with no additional contacts (these are the functions independent of y_j). Any one of the remaining $(2^{p_1 \cdots p_j} - 2^{p_1 \cdots p_{j-1}})$ func-

tions f can be obtained by connecting to the functions $f(y_1, \cdots, y_{j-1}, 1)$, $\cdots, f(y_1, \cdots, y_{j-1}, p_j)$ through $e_1(y_j), \cdots, e_{p_j}(y_j)$ as shown in Fig. 5. In this way a new network is found which produces all functions of the j variables and uses ψ_j contacts where

$$\psi_j - \psi_{j-1} \leq P(2^{p_1 \cdots p_j} - 2^{p_1 \cdots p_{j-1}}). \tag{4}$$

If we now assume that formula (3) holds for ψ_{j-1} we obtain

$$\psi_j \leq P2^{p_1 \cdots p_j}.$$

Thus the theorem will follow by induction when we prove (3) for the case $L = 1$. Since $\psi_0 = 0$ (no contacts are needed to synthesize the two functions 0 and 1) the inequality (4) reduces, when $L = 1$, to

$$\psi_1 \leq P(2^{p_1} - 2)$$

and the theorem is proved.

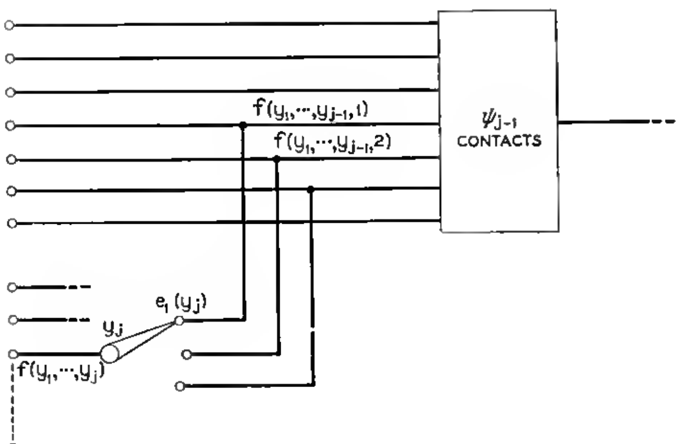

Fig. 5—Network to produce all functions of $(y_1, \ldots, y_j)$.

The induction process we have just described will use up the smallest number of contacts when the large switches are used up first and the small switches last. If, in the process, $p_j > p_{j+1}$, then the number of contacts which would have been saved by making switch y_{j+1} precede switch y_j is found to be

$$(p_j - p_{j+1})2^{p_1 \cdots p_{j-1}}(2^{p_j} - 1)(2^{p_{j+1}} - 1).$$

By adding switches in order of decreasing size in the induction process, the factor P in (3) can be reduced to nearly p_L, the smallest of the L ranges. This refinement is unnecessary for the theory which follows.

The Tree

The number of contacts used in the tree of Fig. 3 is

$$p_1 \cdots p_k \left(1 + \frac{1}{p_k} + \frac{1}{p_{k-1}\, p_k} + \cdots + \frac{1}{p_k \cdots p_2}\right).$$

It can be shown that the most economical way to build the tree is to put the small switches at the narrow end of the tree. If the smallest number of positions of any of the switches in the tree is p_1 then the number of contacts in the tree is less than

$$\frac{p_1 \cdots p_k}{1 - 1/p_1} \leq 2p_1 \cdots p_k. \tag{5}$$

Upper Bound

Having counted the number of contacts which are used in the tree and in the network which produces all functions in Fig. 4, it only remains to decide how many of the given switches $x_1, \cdots, x_M$ are to be put in each of these two parts.

Theorem II. Let P be the largest of the numbers $p_1, \cdots, p_M$ of values which the variables $x_1, \cdots, x_M$ can assume. Then any switching function of $(x_1, \cdots, x_M)$ can be synthesized using no more than

$$(P + \tfrac{1}{2}) \frac{2^{H+1}}{H - 2 \log H} \tag{6}$$

contacts when $H > 4$ bits.

To prove the theorem we consider two cases according as P is greater or less than $H - 2 \log H$.

Case 1: $(P \geq H - 2 \log H)$

In this case we use the synthesis process described above, putting all the switches into the tree and none in the network which produces all functions. The number of contacts used is less than $2 \cdot 2^H$ and the theorem follows because

$$P \geq H - 2 \log H.$$

Case 2: $(P < H - 2 \log H)$

In this case we use the synthesis process described above, putting into the right-hand network a collection S of switches so chosen that $\prod_s p_i$ comes as close as possible to $H - 2 \log H$ without actually exceeding it. Then if

$$\prod_s p_i = (H - 2 \log H)F,$$

we have $F \leq 1$. Also, $F \neq 0$ since any p_i satisfies

$$p_i \leq P < H - 2 \log H.$$

Since $F \neq 0$ it follows that $F > 1/P$. For if $0 < F \leq 1/P$, adding another switch to the collection S will increase $\prod_s p_i$ without making it exceed $H - 2 \log H$.

Using (3) and (5), the number of contacts in the network is less than

$$P2^{(H-2\log H)F} + \frac{2^{H+1}}{(H - 2 \log H)F} \leq \left(\frac{P}{H^2} + \frac{2P}{H - 2 \log H}\right) 2^H.$$

Since $P < H - 2 \log H$,

$$\frac{P}{H^2} < \frac{1}{H} < \frac{1}{H - 2 \log H}$$

and the theorem is proved.

Only a small fraction of the functions will use up this many contacts. In any particular case, the number of contacts used will be about

$$\left(\frac{1}{F} + \frac{1}{2}\right) \frac{2^{H+1}}{H - 2 \log H}$$

and, if many different sizes of switches are used in the network, one should be able to make $1/F$ much closer to 1 than P. Even when all the switches are the same size, one expects

$$\frac{1}{F} < \sqrt{P}$$

in about half the cases.

PART II: N-TERMINAL NETWORKS

Synthesis

Let the accessible terminals be labelled $1, 2, \cdots, N$. To each pair i, j of terminals of an N-terminal network there corresponds a hindrance function $B_{ij}(x_1, \cdots, x_M)$ which tells whether or not there is a closed path between i and j. The B_{ij} satisfies a consistency requirement

(7) $$\text{“}B_{ia} + B_{ab} + \cdots + B_{de} + B_{ej} = 0 \quad \text{implies} \quad B_{ij} = 0\text{”}.$$

The number of consistency requirements (7) is

$$\sum_{r=3}^{N} \frac{N!}{2(N - r)!} \approx \frac{eN!}{2}.$$

However, one can show that all the requirements (7) hold if and only if the $\frac{N(N-1)(N-2)}{2}$ requirements

$$\text{“}B_{ia} + B_{aj} = 0 \quad \text{implies} \quad B_{ij} = 0\text{”}$$

hold.

Conversely, any set of $\frac{N(N-1)}{2}$ hindrance functions which satisfy (7) determine a realizable N-terminal network. One way of synthesizing the network is just to connect, between each pair i, j of terminals, a two-terminal network with hindrance function B_{ij}. It follows from theorem II that

Theorem III. Any N-terminal switching function of switches with P or fewer positions can be synthesized with no more than

$$N(N-1)\left(P + \frac{1}{2}\right)\frac{2^H}{H - 2\log H}$$

contacts when $H > 4$ bits.

The network can also be synthesized using N trees, each of which produces all of the possible functions $e_r(x_1) + e_s(x_2) + \cdots + e_s(x_M)$. Each terminal is connected to the input lead of one of the trees; and the output leads, to which the terminals are connected in any given state $(x_1, \cdots, x_M)$, are interconnected in the way one wants the terminals to be interconnected in that state. The number of contacts used in this type of synthesis is less than

$$N2^{H+1}.$$

The synthesis using two-terminal networks ordinarily requires fewer contacts than the one using trees as long as

$$H - 2\log H > \tfrac{1}{2}(N-1)(P + \tfrac{1}{2})$$

An example illustrating the design of a typical three-terminal network is given in the appendix.

Number of Functions

Every N-terminal switching function determines a realizable matrix of hindrance functions $B_{ij}(x_1, \cdots, x_M)$. It is important to know the number of different switching functions of $(x_1, \cdots, x_M)$.

A state of the N terminals is determined by specifying the groups of terminals which are connected together. The number $\phi(N)$ of such states is the number of ways that N different objects can be distributed into $1, 2, \cdots$, or N parcels when the parcels are indistinguishable from one another and no parcel is left empty.

A switching function represents one of these $\phi(N)$ different states for each of the 2^H different switch settings. Hence the number of switching functions is

$$(\phi(N))^{2^H}.$$

Although there is no simple formula for $\phi(N)$, a generating function for $\phi(N)$ is well known:[4]

$$e^{e^z-1} = \sum_{n=0}^{\infty} \frac{\phi(n)}{n!} z^n. \tag{8}$$

A recursion formula which can be used to calculate $\phi(N)$ is

$$\phi(N + 1) = \sum_{k=0}^{N} C_{N,k}\,\phi(k). \tag{9}$$

When N is large $\phi(N)$ can be estimated with the help of the upper and lower bounds to be derived. These bounds will be of use to us later mainly because they show that, for large N, $\log \phi(N)$ is approximately $N \log N$.

Theorem IV.

$$\phi(N) \leq \frac{N!}{e} \frac{e^{N/\log_e N}}{\left[\log_e \dfrac{N}{\log_e N}\right]^N}. \tag{10}$$

Proof. The maximum value of $| e^{e^z-1} |$ on the circle $| z | = r$ is e^{e^r-1}. Using (8) and Cauchy's inequality for the N^{th} coefficient in a power series,

$$\phi(N) \leq \frac{N!\,e^{e^r-1}}{r^N} \tag{11}$$

for all $r > 0$. The best estimate of $\phi(N)$ will be obtained by minimizing (11) on r. To do this one sets $r = r_0$ where

$$r_0 e^{r_0} = N.$$

The simpler result (10) is obtained from (11) by setting

$$r = \log_e \frac{N}{\log_e N}$$

Theorem V.

$$\frac{A^N}{A!} \leq \phi(N) \quad \textit{for all integers } A. \tag{12}$$

Proof. Let $Q(N, A)$ be the number of ways that N different objects can be distributed into $1, 2, \cdots$, or A indistinguishable parcels. Then $Q(N, A)A!$

[4] W. A. Whitworth, Choice and Chance, p. 88, Cambridge, Bell, 1901.

must be greater than the number of ways N different objects can be placed in A different boxes (labelled $1, \cdots, A$); i.e.

$$A^N \leq Q(N, A)A! \leq \phi(N)A!.$$

To obtain the best lower bound from (12) one may maximize on A. The best value of A to use is one which comes close to satisfying

$$\left(1 + \frac{1}{A}\right)^{N-1} = A.$$

For large N the solution is approximately

$$A = \frac{N}{\log_e N}.$$

To perform the minimization in theorem IV more carefully one would solve

$$A_0 \log_e A_0 = \mathrm{N}$$

for A_0. This is the minimizing equation given in theorem IV with $r_0 = \log_e A_0$. It is also very nearly the minimizing equation given in theorem V.

Then our proofs of theorems IV and V show that

$$\frac{A_0^N}{A_0!} \leq \phi(N) \leq N! \frac{e^{A_0-1}}{(\log_e A_0)^N}.$$

For large N these bounds differ by a factor of about

$$\frac{2\pi}{e} \frac{N}{\sqrt{\log_e N}}$$

More accurate information about the behavior of $\phi(N)$ for large N is provided by an asymptotic series found by L. F. Epstein.[5] The first term in his series is

$$\phi(N) \sim \frac{a^{N-a} e^{a-1}}{\sqrt{\log_e a}}$$

where a is found by solving

$$a \log_e (a + 1) = N.$$

Figure 6 is a graph of $\phi(N)$ vs. N using a log log scale for $\phi(N)$. The points are exact values and the curves show the upper and lower bounds.

[5] L. F. Epstein, *J.M.P.*, *18*, 3, pp. 153–173 (1939).

Number of Graphs

Let $G(N, K)$ be the number of topologically distinct linear graphs which can be drawn interconnecting the N-terminals and using K branches. $G(N, K)$ counts *all* graphs including graphs with dangling branches and disconnected pieces. It also counts graphs in which any or all of the N-terminals are connected to no branches. Figures 7a, b, c, d, e show some topologically distinct graphs which would be counted in finding $G(3, 10)$.

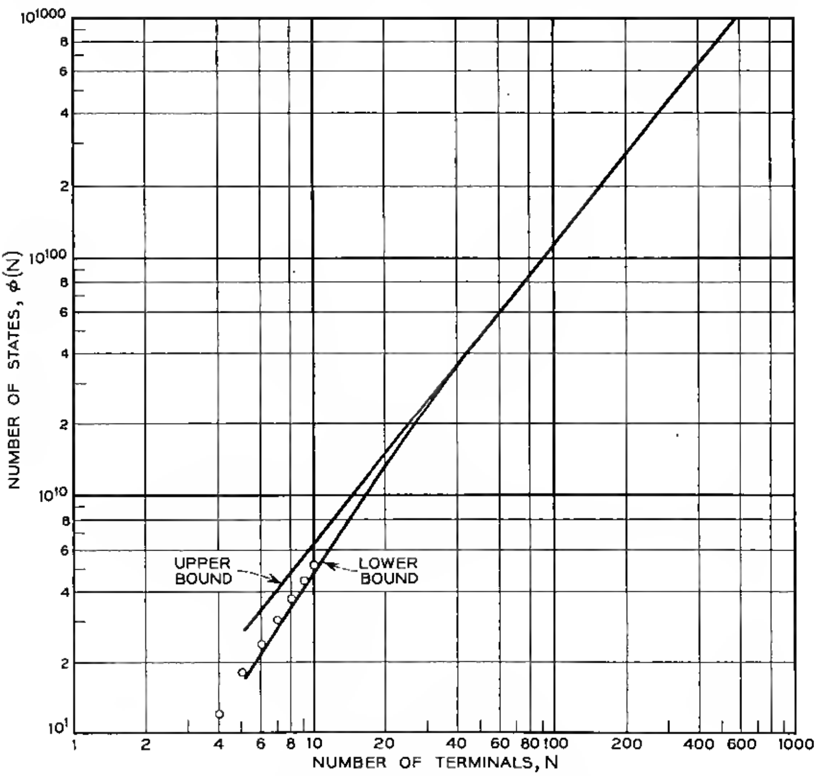

Fig. 6—Number of possible states of N terminals.

Graph 7f is topologically identical with graph 7b and so is not to be counted again. The first step toward finding a lower bound on the number of contacts which almost all switching functions require is to find an upper bound on $G(N, K)$.

Theorem VI.

$$G(N, K) < 2^{N+K}(N + 2K)^{K} .$$

Proof. Every linear graph can be constructed hy the following process. Let the branches be numbered 1, 2, $\cdots$, K and let the end points of the k^{th} branch be called A_k and B_k. There are $K - 1$ places where partition marks can be inserted in the sequence $A_1, \cdots, A_k$ and hence there are 2^{K-1} ways of partitioning the A_k's into groups of the form

$$G_1 = (A_1, A_2, \cdots, A_a)$$
$$G_2 = (A_{a+1}, \cdots, A_b)$$
$$G_3 = (A_{b+1}, \cdots, A_c)$$
$$\vdots \qquad \vdots$$

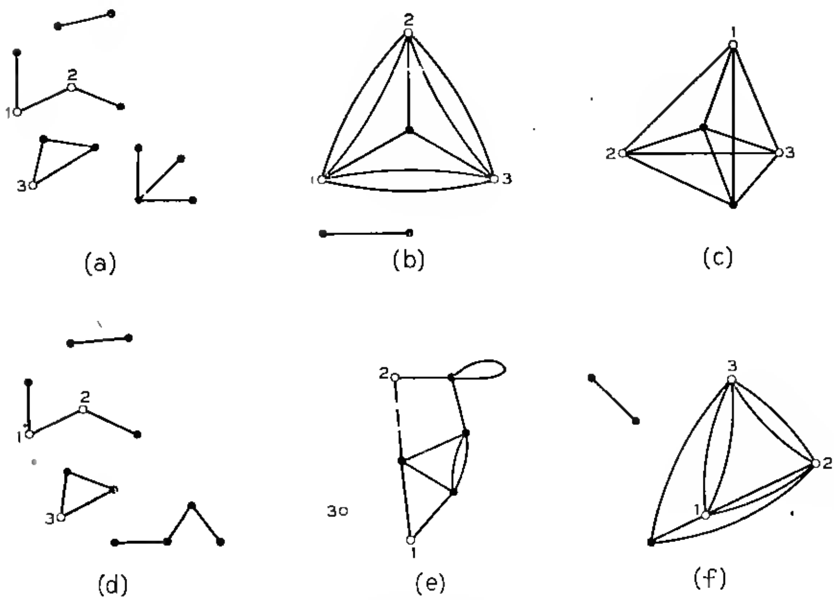

Fig. 7—Examples of graphs.

There are 2^N ways of selecting some of the terminals 1, 2 $\cdots$, N. Suppose that m of the terminals have been selected; then pick one of the partitions of the A_k's which has m or more groups $G_1, \cdots, G_{m+s}$. Connect all the end points in G_1 to the first selected terminal, all the end points in G_2 to the second selected terminal, etc. Next connect the terminals in $G_{m+1}, \cdots,$ G_{m+s} together to form s nodes. The numher of ways of performing all these operations is less than

$$2^{N+K-1}.$$

Connect B_1 to one of the N-terminals or to one of the nodes just made or else use B_1 to make a new node. Connect B_2 to one of the terminals or nodes or else use B_2 to make a new node, etc. The number of ways of connecting $B_1, \cdots, B_K$ is less than

$$(N + K + 1)(N + K + 2) \cdots (N + 2K) < (N + 2K)^K,$$

which proves the theorem.

Since most graphs can be constructed in many different ways by this process, theorem VI gives a very poor estimate of $G(N, K)$. In the application which we will make of $G(N, K)$ it is enough to know that $\log G(N, K)$ behaves something like $K \log K$. To prove that $K \log K$ cannot be replaced by anything much smaller we now give a lower bound for $G(N, K)$.

Theorem VII.

$$\frac{(\phi(K))^2}{2^K K!} \leq G(N, K)$$

Proof. $G(N, K)$ is larger than the number of graphs which can be drawn without specifying certain nodes as terminals $1, 2, \cdots, N$. Of these graphs let us count only those which have the property that no cycle in the graph has an odd number of branches. Another characterization of these graphs is that their nodes can be divided into two classes A and B such that no branch joins two nodes of the same class.

To construct such graphs we first number the branches $1, 2, \cdots, K$ and give them an orientation (say by putting an arrow head at one end of each branch). The front ends of the branches can be grouped together into nodes in $\phi(K)$ ways. Then the tail ends of the branches are grouped together in one of $\phi(K)$ ways. In this way a total of $(\phi(K))^2$ different graphs can be drawn, in which the branches are numbered and oriented. If we now ignore the numbers on the branches we still have at least $\frac{(\phi(K))^2}{K!}$ distinct graphs with oriented branches. If the orientation is ignored, the number of topologically different graphs which remain is greater than

$$\frac{(\phi(K))^2}{2^K K!}.$$

Lower Bound

We have seen that any switching function can be realized with no more than about

$$\frac{N^2 P 2^H}{H - 2 \log H}$$

contacts. To show that this number cannot be improved very much we will now show that almost all switching functions require a number of contacts of this order of magnitude.

Theorem VIII. Let any $\epsilon > 0$ be given. The fraction of switching functions which can be synthesized using less than

$$(1 - \epsilon)\ \frac{2^H \log \phi(N)}{H + \log \log \phi(N)} \tag{13}$$

contacts approaches zero uniformly as the number M of switches becomes large.[6]

Proof. The number of switching functions which can be constructed with K contacts or less is certainly smaller than the number of ways the K branches of the $G(N, K)$ graphs can be replaced by contacts $e_r(x_i)$ or open circuits; i.e. smaller than

$$G(N, K) \left(\sum_{i=1}^{M} p_i + 1 \right)^K.$$

By theorem VI the fraction $F(K)$ of the $(\phi(N))^{2^H}$ switching functions which can be built using K contacts or less satisfies

$$F(K) \leq 2^{N+K}(N + 2K)^K \left(\sum_{i=1}^{M} p_i + 1 \right)^K (\phi(N))^{-2^H}$$

$$\leq 2^{2N+K\left(\log K + 2 + \log\left[\sum_{i=1}^{M} p_i + 1\right]\right) - 2^H \log \phi}$$

where we have used

$$\log_2 (N + 2K) \leq \log_2 2K + \frac{N}{2K} \log_2 e$$

$$\leq \log_2 2K + \frac{N}{K}.$$

When K is the expression (13), one finds

$$F(K) \leq 2^{2N}(\phi(N))^{2^H\left(-\epsilon + \frac{\log(\Sigma p+1)+2}{H+\log\log\phi}\right)}.$$

Since $\frac{\log \Sigma p_i}{H} = \frac{\log \Sigma p_i}{\log \Pi p_i}$ approaches zero as the number of switches M gets large, it follows that for sufficiently large M and any N

[6] The word *uniformly* is used to indicate that the fraction in question can be made smaller than any given number $\delta > 0$ by making M larger than a certain number $M(\delta)$ which depends on δ but *not on* N.

$$\frac{\log(\Sigma p_i + 1) + 2}{H + \log\log\phi(N)} < \frac{\epsilon}{2}. \tag{14}$$

Then

$$F(K) \leq 2^{2N}(\phi(N))^{-\epsilon 2^{H-1}} \tag{15}$$

which approaches zero uniformly as M increases.

For most of the switching functions of practical interest H is much bigger than $\log\log\phi(N)$. In these cases the number

$$\frac{N^2 P 2^H}{H - 2\log H}$$

is larger than (13) by a factor of about $NP/\log N$. In the case of two-terminal relay circuits the corresponding factor found by Shannon was only 8. It is not clear whether this difference indicates that there is a wider range of complexity for N-terminal networks than for two-terminal networks or that our methods for obtaining upper and lower bounds lose some of their effectiveness as N increases. Nevertheless, (13) is surprisingly large, as we shall see in the example which follows.

Example. Consider a telephone central office with 10,000 lines. If the office must be able to connect the lines together in pairs in any arrangement and to remember which line of a pair originated the call, a count of the number of different states which must be produced reveals that the office needs a memory of at least $H = 64{,}000$ bits, which can be supplied using 19,200 switches with 10 positions each. The number of other switching functions that one might ask these 19,200 switches to perform is

$$\phi(10{,}000)^{2^{64{,}000}} = (10^{30{,}000})^{10^{19{,}200}} = 10^{10^{19{,}200}} \text{ approx.}$$

To apply theorem VIII to these other functions we first note that (14) will be satisfied as long as we pick ϵ greater than .006. Then, substituting in (13) and (15) we discover that the chance that one of these switching functions chosen at random can be synthesized with less than about

$$10^{19{,}000} \text{ contacts}$$

is less than some number of the order of

$$10^{-10^{19{,}200}}.$$

If the same calculation is repeated for a 10,000-line office which is capable

of handling only 1,250 calls at one time we find

$$
\begin{aligned}
H &= 22{,}000 \text{ bits,} \\
\# \text{ switches} &= 6{,}666 \\
\# \text{ functions} &= 10^{10^{6.670}} \\
\epsilon &= .015 \text{ or larger} \\
\# \text{ contacts} &= 10^{6.671} \text{ or more for all but a fraction } 10^{-10^{6.668}} \text{ of the functions.}
\end{aligned}
$$

Although these numbers of contacts appear incredible at first sight, there is no reason to expect the number of contacts for almost all switching functions to be a good indication of the number of contacts needed for the switching functions of practical use. This phenomenon has been discussed in detail by Shannon for the case of two-terminal networks. For N-terminal networks there are at least two other factors which may be mentioned.

Almost all switching functions can assume states which are not typical of the functions encountered in practice. For example, it can be shown that of the $\phi(N)$ possible ways of distributing N different things into parcels, almost all of them use a number of parcels which is near $N/\log_e N$. Thus, in a typical state of a typical switching function the terminals are connected together in groups which average about $\log_e N$ terminals per group in size. Telephone switching equipment ordinarily connects terminals together in pairs or in small groups.

A big difference between the design of two-terminal networks and of N-terminal networks is that, in the former case, one wants to obtain one specific switching function while, in the latter case, one is usually satisfied with a network which can produce certain desired states. There are many switching functions which all produce the same states but for different settings of the switches. For example, if the 2^H desired states are all different, the designer will be content with any one of $(2^H)!$ different switching functions. We believe that it actually would require something like $10^{6.671}$ contacts to build a central office if the designer first listed all the desired states at random $S_1, S_2, \cdots, S_{2^H}$ and then required the office to be in state S_1 for switch setting $(1, 1, \cdots, 1)$, in S_2 for the switch setting $(1, 1, \cdots, 2)$, etc.

Number of Selector Switch Rotors

Since our estimate (13) of the number of contacts is the same whether the memory H is stored in two-position switches or in larger selector switches, one might hope that the selector switch circuits could be built using fewer rotors than the corresponding two-position switch circuits. We believe that this is not true. A typical node in one of the graphs constructed by the process of theorem VI has only about three or four branches connected to

it. This is so regardless of how large K is. If the rotor of a selector switch is connected to such a node, the chance is great that none of the other branches at the node are operated by this same switching variable. Hence we suspect that a typical switching network requires almost as mnay rotors as contacts.

Acknowledgment

The author has discussed this switching problem with B. D. Holbrook, A. W. Horton, J. Riordan, and C. E. Shannon and has received valuable comments and ideas from them. He is also indebted to A. E. Joel and W. Keister for their suggestions for improving the readability of the manuscript.

Table I

x	y	z	$f(x, y, z)$	f_{12}	f_{13}	f_{23}
0	0	1	(12) (3)	0	1	1
0	0	2	(123)	0	0	0
0	0	3	(1) (2) (3)	1	1	1
0	0	4	(13) (2)	1	0	1
0	1	1	(13) (2)	1	0	1
0	1	2	(1) (23)	1	1	0
0	1	3	(123)	0	0	0
0	1	4	(1) (23)	1	1	0
1	0	1	(1) (23)	1	1	0
1	0	2	(13) (2)	1	0	1
1	0	3	(1) (2) (3)	1	1	1
1	0	4	(12) (3)	0	1	1
1	1	1	(123)	0	0	0
1	1	2	(123)	0	0	0
1	1	3	(1) (23)	1	1	0
1	1	4	(1) (2) (3)	1	1	1

APPENDIX

To illustrate how the network synthesis method operates in a typical case consider a three-terminal network using three switches x, y, z. Switches x and y have two positions 0 and 1, and z has four positions 1, 2, 3, 4. A three-terminal switching function $f(x, y, z)$ is defined by means of the first four columns of Table I. The sixteen entries in column four represent the states of the terminals which the network must produce for the corresponding switch settings given in the columns labelled x, y, z. In column four, parentheses are used to group terminals which are connected together; for example $f(1, 0, 4)$ is the state in which terminals 1 and 2 are connected together and 3 is left free.

A network with switching function $f(x, y, z)$ will be designed by connecting two-terminal networks between the pairs of terminals 1, 2; 2, 3; and 1, 3. The hindrance functions of these three two-terminal networks will be called

f_{12}, f_{23}, and f_{13}. We determine them, as shown in the last three columns of Table I, by setting $f_{ij} = 0$ whenever terminals i and j are to be connected and $f_{ij} = 1$ otherwise. Our methods of two-terminal network synthesis will produce the f_{ij} networks.

Our criterion ($P > H - \log H$) for deciding which of the two synthesis methods to use (case 1 or case 2 of theorem II) is not the best rule when H is as small as it is in this example. By actually trying the different ways of apportioning switches with 2, 2, and 4 positions between a tree and a network to produce all functions, one finds that the most economical way is to

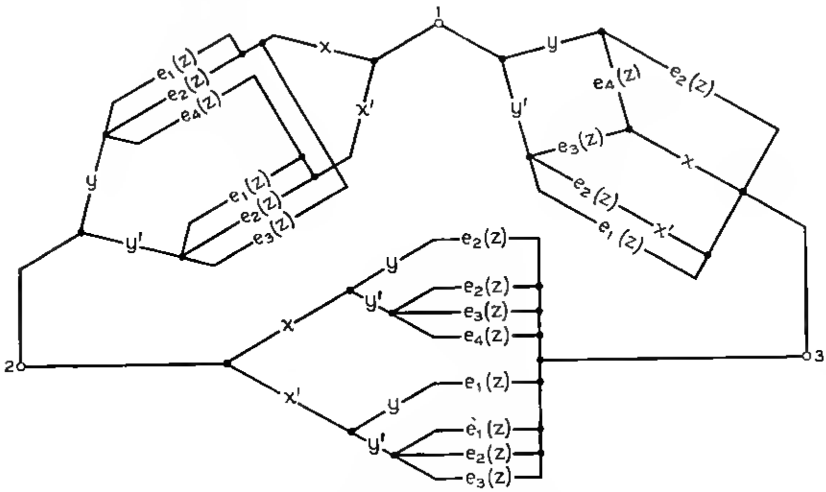

Fig. 8—Network with the 3-terminal switching function $f(x, y, z)$ of Table I.

put a two-position switch, say x, into a network which provides all functions of x (0, 1, x, and x'), and the other switches into a tree. When this procedure is adopted we next express f_{ij} in the form of identity (2). For example

$$f_{13} = [y + e_2(z)][y + e_4(z) + x][y' + e_3(z) + x][y' + e_2(z) + x'][y' + e_1(z)].$$

The synthesis method described in the text then leads directly to a network for f_{13} which is shown joining terminals 1 and 3 in Fig. 8. The network for f_{12} which is shown in Fig. 8 was obtained by the same process. For the sake of illustration the f_{23} network was found using a tree only.